METAMORPHOSIS OF PASSWORD

Sumeeha V Ashraf

MES20MCA-2054

Product Owner : Mr. Nowshad C.V

TABLE OF CONTENTS

PROJECT DESCRIPTION

- The project is to develop a password transmission scheme by rasterizing the characters in the password and sending it in different byte order based on generated random number.
- Raster is 8 * 8 dots. That is 8 bytes.
- Dots for each character is stored as 8 bytes in a random file.
- Password is never sent to the server.
- During both registering and login, rasterized image is sent to the server.
- The order in which the bytes are sent will differ according to a random number.
- On entering the password, the client calculates the order in which, client has to send the bytes, based on a random number.
- The server arranges the byte received into scrambled order, in both cases of registering and logging in based on the information stored in the upper nibble of each byte.
- During registration, the bytes are written into a file, the name of which is the user id. On login, the received bytes are compared with the bytes read from the file.

DEVELOPING ENVIRONMENT

- Operating system: Windows 10
- Language: Java (version : 17.0.1)

METHODOLOGY

- In order to rasterize passwords, initially a raster file named chardot. An application named Charshape is created for this purpose. It provides an eight into eight matrix in which the shape of the character is created by 1 and zeroes. Bits in a row is combined into one byte. Thus 8 bytes represent a character.
- Only characters from ASCII 48 to 127 is used.
- For login and registration, a client and a server softwares are created. The client accepts user id and password. Two buttons Login and registration are provided. On pressing either of the button, id is send to the server.
- If it is registration, the server checks whether the user has already registered and if so the existing password is changed. Then it gives OK message.
- If it is Login, the server checks whether the user has already registered and if not it sends error message. Otherwise it gives OK message. On receiving OK message, client sends the password.
- The order in which the bytes are sent will differ according to a generated random number and 2 bytes are send by breaking into nibbles and adding recovery information with the nibbles to make them two separate bytes.
- Mudulo 8 of the random number is calculated. The byte pair order will be based on the mode 2 of another random number generated. When the modulo 8 number reaches 8 , it will be made 0.

- Creation of the byte pair is done by breaking the byte to 2 nibbles and storing both nibbles as lower nibbles of each byte. The byte corresponding to lower nibble is ORed with the original byte order multiplied by 16 to occupy the most significant 4 bits.
- Similarly the byte corresponding to upper Nibble is padded left with 1000. This makes both bytes self identifiable. The order within the pair is decided on the parity of the current second (Odd or even) .
- If it is odd the byte representing upper nibble will be sent first.
- Since position identifying information is built in, the server can recover the original bytes represented by the raster image.
- On registration, the server creates a file with name as the userid and stores the raster of the password, after restoration to the original, in this file.
- On login the received password, after restoration, is compared to the password stored in the file.

FUTURE ENHANCEMENT

- Future release can be modified to by storing the dot structure of each characters in the server side and perform the transmission of password by rasterizing based on the random number generated.
- When the program starts the dots are transferred to cashe.
- This can reduce the risk of storing the dot structure in the file and thus enhance performance of the password transmission scheme.

PROJECT PLAN

ID	Task Name	Start Date	End Date	Days	Status
1	Sprint 1	22/12/2021	03/01/2022	13	Completed
2	Sprint 2	11/01/2022	24/01/2022	14	Completed
3	Sprint 3	07/02/2022	20/02/2022	14	Completed

PRODUCT BACKLOG

User story ID	Priority <High/Medium/Low>	Size (Hours)	Sprint <#>	Status <Planned/In progress/Completed>	Release Date	Release Goal
1	Medium	12	1	Completed	11/01/2022	Character Design
2	High	18	2	Completed	24/01/2022	Front end
3	High	20	3	Completed	20/02/2022	Back end

SPRINT BACKLOG PLAN

Sprint 1 :-

Backlog Item	Status & completion date	Original estimate in hours	Day1 25/12	Day2 26/12	Day3 27/12	Day4 28/12	Day5 29/12	Day6 30/12	Day7 31/12	Day8 01/01	Day9 02/01	Day10 03/01	Day11 04/01	Day12 05/01	Day13 06/01	Day14 07/01
UI designing	25/12/2021	6	1	0	3	2	0	0	0	0	0	0	0	0	0	0
Coding	01/01/2022	12	0	0	0	0	3	2	0	3	0	1	3	0	0	0
Testing	03/01/2022	4	0	0	0	0	0	0	0	0	0	0	0	2	2	0
TOTAL		**22**	**1**	**0**	**3**	**2**	**3**	**2**	**0**	**3**	**0**	**1**	**3**	**2**	**2**	**0**

SPRINT BACKLOG PLAN

Sprint 2 :-

Backlog Item	Status & completion date	Original estimate in hours	Day1 11/01	Day2 12/01	Day3 13/01	Day4 14/01	Day5 15/01	Day6 16/01	Day7 17/01	Day8 18/01	Day9 19/01	Day10 20/01	Day11 21/01	Day12 22/01	Day13 23/01	Day14 24/01
UI designing	11/01/2022	4	1	0	1	2	0	0	0	0	0	0	0	0	0	0
Coding	15/01/2022	10	0	0	0	0	2	1	0	3	1	1	2	0	0	0
Testing	24/01/2022	4	0	0	0	0	0	0	0	0	0	0	0	2	2	0
TOTAL		**18**	**1**	**0**	**1**	**2**	**2**	**1**	**0**	**3**	**1**	**1**	**2**	**2**	**2**	**0**

SPRINT BACKLOG PLAN

Sprint 3 :-

Backlog Item	Status & completion date	Original estimate in hours	Day1 07/02	Day2 08/02	Day3 09/02	Day4 10/02	Day5 11/02	Day6 12/02	Day7 13/02	Day8 14/02	Day9 15/02	Day10 16/02	Day11 17/02	Day12 18/02	Day13 19/02	Day14 20/02
Coding	07/02/2022	15	2	3	0	1	1	3	0	3	1	1	2	0	0	0
Testing	20/02/2022	5	0	0	0	0	0	0	0	0	0	0	0	2	2	1
TOTAL		**20**	**2**	**3**	**0**	**1**	**1**	**3**	**0**	**3**	**1**	**1**	**2**	**2**	**2**	**1**

SPRINT BACKLOG ACTUAL

Sprint 1 :-

Backlog Item	Status & completion date	Original estimate in hours	Day1 25/12	Day2 26/12	Day3 27/12	Day4 28/12	Day5 29/12	Day6 30/12	Day7 31/12	Day8 01/01	Day9 02/01	Day10 03/01	Day11 04/01	Day12 05/01	Day13 06/01	Day14 07/01
UI designing	25/12/2021	6	1	0	3	2	0	0	0	0	0	0	0	0	0	0
Coding	01/01/2022	12	0	0	0	0	3	2	0	3	0	1	3	0	0	0
Testing	03/01/2022	4	0	0	0	0	0	0	0	0	0	0	0	2	2	0
TOTAL		**22**	**1**	**0**	**3**	**2**	**3**	**2**	**0**	**3**	**0**	**1**	**3**	**2**	**2**	**0**

SPRINT BACKLOG ACTUAL

Sprint 2 :-

Backlog Item	Status & completion date	Original estimate in hours	Day1 11/01	Day2 12/01	Day3 13/01	Day4 14/01	Day5 15/01	Day6 16/01	Day7 17/01	Day8 18/01	Day9 19/01	Day10 20/01	Day11 21/01	Day12 22/01	Day13 23/01	Day14 24/01
UI designing	11/01/2022	4	1	0	1	2	0	0	0	0	0	0	0	0	0	0
Coding	15/01/2022	10	0	0	0	0	2	1	0	3	1	1	2	0	0	0
Testing	24/01/2022	4	0	0	0	0	0	0	0	0	0	0	0	2	2	0
TOTAL		**18**	**1**	**0**	**1**	**2**	**2**	**1**	**0**	**3**	**1**	**1**	**2**	**2**	**2**	**0**

SPRINT BACKLOG ACTUAL

Sprint 3 :-

Backlog Item	Status & completion date	Original estimate in hours	Day1 07/02	Day2 08/02	Day3 09/02	Day4 10/02	Day5 11/02	Day6 12/02	Day7 13/02	Day8 14/02	Day9 15/02	Day10 16/02	Day11 17/02	Day12 18/02	Day13 19/02	Day14 20/02
Coding	07/02/2022	15	2	3	0	1	1	3	0	3	1	1	2	0	0	0
Testing	20/02/2022	5	0	0	0	0	0	0	0	0	0	0	0	2	2	1
TOTAL		**20**	**2**	**3**	**0**	**1**	**1**	**3**	**0**	**3**	**1**	**1**	**2**	**2**	**2**	**1**

Thanks!